Snowflake Prototyping

Loy Sikdar

2025

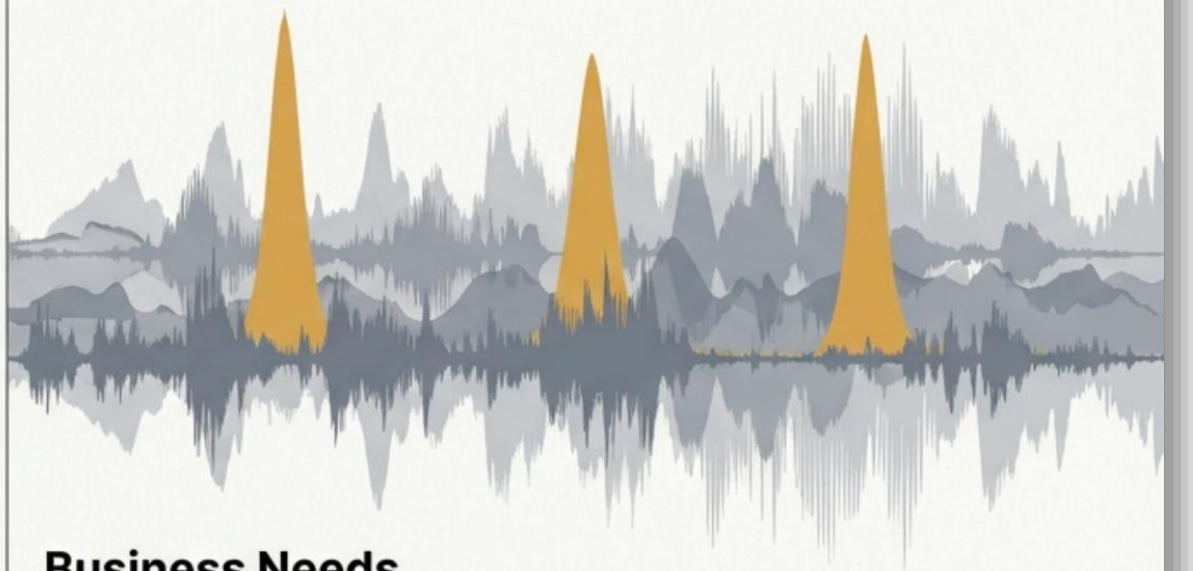
End-to-End Anomaly Detection & AI-Powered Explanation in Snowflake

A technical walkthrough of a call center solution built with Snowpark, Cortex, and `mistral-large2`.

The Supervisor's Challenge: Finding and Understanding Outliers at Scale

Core Problem Description

Call centers generate massive volumes of interaction data. Within this data, critical outliers—such as calls with extremely long handle times, unusual knowledge management (KM) usage, or repeated escalations—are difficult to systematically identify and diagnose.



Business Needs

- ✓ **Automated Detection:** A scalable method to flag anomalous calls without manual review.
- ✓ **Root Cause Understanding:** A clear way to understand *why* a call was flagged as anomalous.
- ✓ **Actionable Insights:** Guidance on what actions to take, from agent coaching to process improvement.

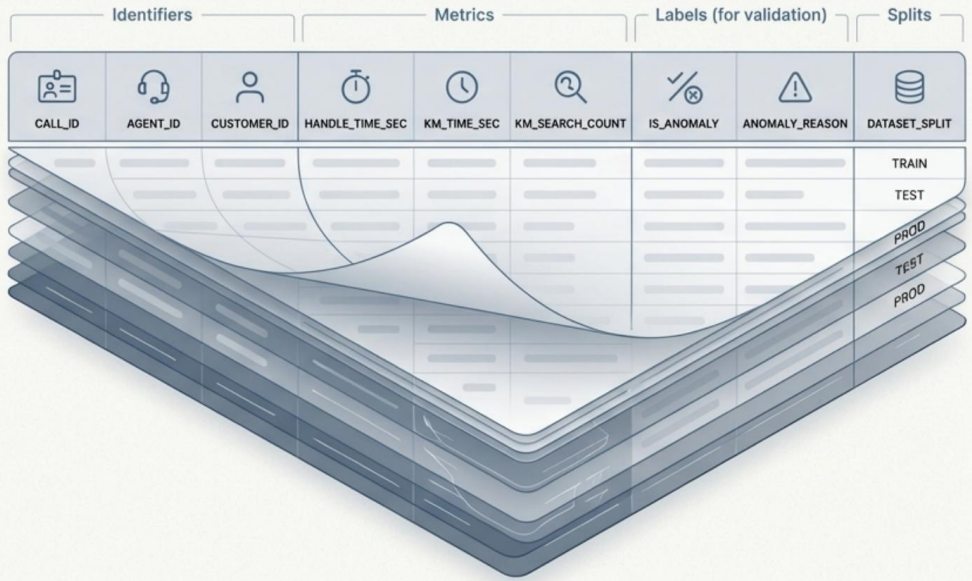
Building the Foundation: A Synthetic Call Center Dataset

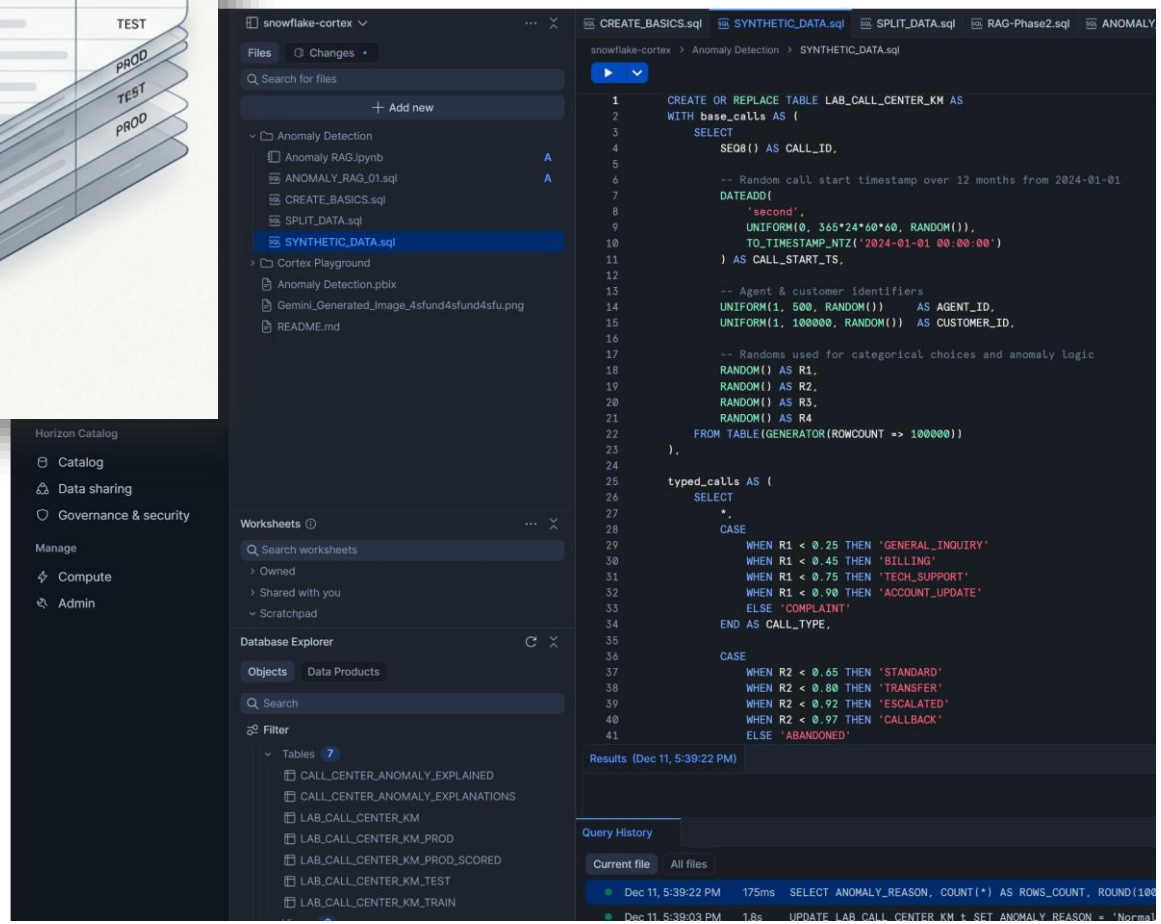
Rationale

To develop and validate our models without using real customer data, we engineered a synthetic dataset simulating 12 months of activity (~100,000 call records).

Core Table

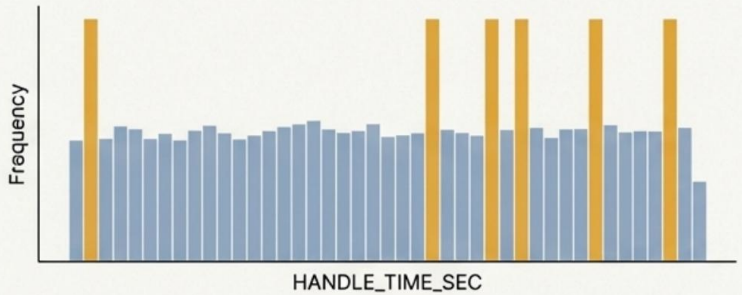
`LAB_CALL_CENTER_KM`





Engineering a Labeled Dataset for Model Development

Anomaly Injection Strategy



We intentionally label ~10% of calls as anomalous with `ANOMALY_REASON = 'ExtremeLongHandleTime''.

These records have massively inflated `HANDLE_TIME_SEC` values compared to the 'Normal' population.

This creates a clean, labeled ground-truth set for evaluating model performance.

Time-Based Data Splitting

The `DATASET_SPLIT` column partitions the data chronologically using `CALL_DATE`.

This structure supports realistic model training, validation, and simulated production scoring.

owflake-cortex › Anomaly Detection › SPLIT_DATA.sql

```sql
USE WAREHOUSE ANOM_DETX_WH;
USE SCHEMA ANOM_DETX.AD_LABS;

ALTER TABLE LAB_CALL_CENTER_KM
ADD COLUMN IF NOT EXISTS DATASET_SPLIT STRING;


-- Label each row as TRAIN / TEST / PROD

UPDATE LAB_CALL_CENTER_KM
SET DATASET_SPLIT =
    CASE
        WHEN CALL_DATE < '2024-09-01' THEN 'TRAIN'          -- Jan-Aug
        WHEN CALL_DATE < '2024-11-01' THEN 'TEST'           -- Sep-Oct
        ELSE 'PROD'                                         -- Nov-Dec
    END;

-- Create physical tables from the split
CREATE OR REPLACE TABLE LAB_CALL_CENTER_KM_TRAIN AS
SELECT *
FROM LAB_CALL_CENTER_KM
WHERE DATASET_SPLIT = 'TRAIN';

CREATE OR REPLACE TABLE LAB_CALL_CENTER_KM_TEST AS
SELECT *
FROM LAB_CALL_CENTER_KM
WHERE DATASET_SPLIT = 'TEST';

CREATE OR REPLACE TABLE LAB_CALL_CENTER_KM_PROD AS
SELECT *
FROM LAB_CALL_CENTER_KM
WHERE DATASET_SPLIT = 'PROD';

```

README.md

Worksheets

Search worksheets

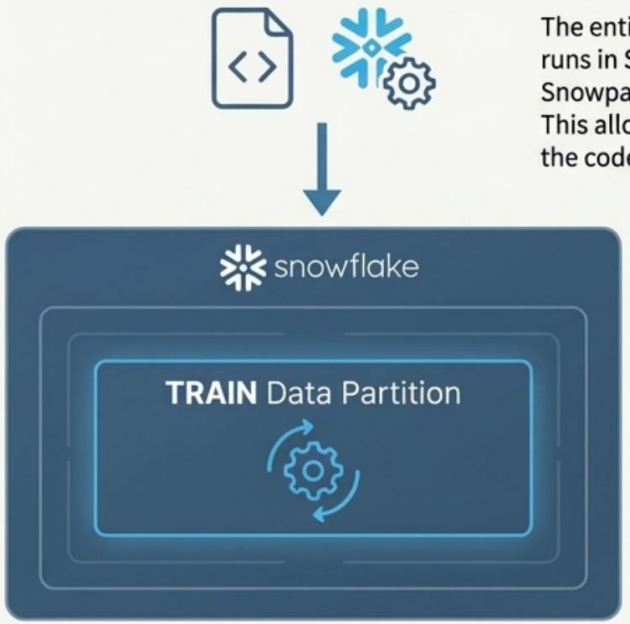
Owned

Shared with you

Scratchpad

Training the Detection Model with Snowpark and Scikit-learn

Code to Data Flow



The entire ML workflow runs in Snowflake using Snowpark for Python. This allows us to bring the code to the data.

Section 1: Feature Selection

We use a numeric feature set derived from LAB_CALL_CENTER_KM , including:

- HANDLE_TIME_SEC
- KM_TIME_SEC
- AFTER_CALL_WORK_SEC
- HOLD_TIME_SEC
- KM_SEARCH_COUNT
- AGENT_TENURE_MONTHS
- HOUR_OF_DAY

Section 2: Algorithm

We use an **Isolation Forest**, an unsupervised algorithm well-suited for anomaly detection.

- **Key Parameter**: `contamination=0.10`, set to align with our known 10% anomaly injection rate.

The model is fit exclusively on the TRAIN data partition.

```
n ∨ as cell5
1  from sklearn.ensemble import IsolationForest
2
3  # contamination ~ expected anomaly rate (10% from your synthetic design)
4  iso = IsolationForest(
5      n_estimators=200,
6      contamination=0.10,
7      random_state=42,
8  )
9
10 iso.fit(X_train)
11
```

```
ationForest(contamination=0.1, n_estimators=200, random_state=42)
```

```
down ∨ as cell13
```

Evaluate on TEST (using your IS_ANOMALY labels)

```
n ∨ as cell6
1  from sklearn.metrics import classification_report, roc_auc_score
2
3  # IsolationForest: -1 = anomaly, 1 = normal
4  test_preds = iso.predict(X_test)
5  test_scores = iso.decision_function(X_test)  # higher = more normal
6
7  # Convert to 0/1 anomaly prediction
8  test_pred_is_anom = (test_preds == -1).astype(int)
9
10 print("Classification report (TEST):")
11 print(classification_report(y_test, test_pred_is_anom, digits=3))
12
13 # AUC using anomaly scores (flip sign so higher score = more anomalous)
14 test_auc = roc_auc_score(y_test, -test_scores)
15 print("ROC AUC (TEST):", round(test_auc, 3))
16
```

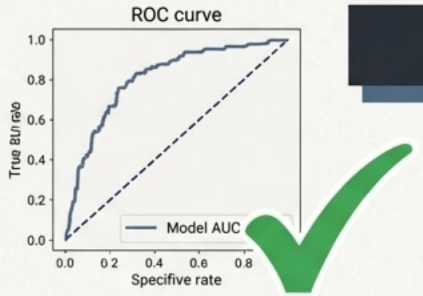
```
Classification report (TEST):
              precision    recall  f1-score   support

           0      0.901     0.905     0.903     14995
           1      0.129     0.124     0.126      1708

    accuracy                         0.825     16703
   macro avg      0.515     0.514     0.514     16703
weighted avg      0.822     0.825     0.823     16703

ROC AUC (TEST): 0.532
```

Evaluating Performance and Scoring Production Data

Evaluation on TEST Data

We score the `TEST` partition and compare the model's predictions against the ground-truth `IS_ANOMALY` labels.

	Precision	Recall	F1-Score
0	0.00	0.70	0.90
1	0.00	0.70	0.70
Linear Score	Precision	0.85	0.88
	Standard	0.50	0.50



Performance is measured using a classification report (precision, recall, F1) and ROC AUC, confirming the model effectively identifies the synthetic anomalies.

Scoring PROD Data & Persisting Results

The validated model is applied to the `PROD` data partition. The results, `PRED_IS_ANOMALY` (0/1) and `ANOMALY_SCORE` (a continuous value), are written back to a new table:

`LAB_CALL_CENTER_KM_PROD_SCORED`

```
edict(X_prod)
cision_function(X_prod)   # higher = more normal

prod_pd["PRED_IS_ANOMALY"] = (prod_preds == -1).astype(int)
prod_pd["ANOMALY_SCORE"]   = -prod_scores    # higher = more anomalous

# Convert back to Snowpark DF
scored_prod_df = session.create_dataframe(prod_pd)

# Persist as a new table
scored_prod_df.write.save_as_table(
    "LAB_CALL_CENTER_KM_PROD_SCORED",
    mode="overwrite"
)
```

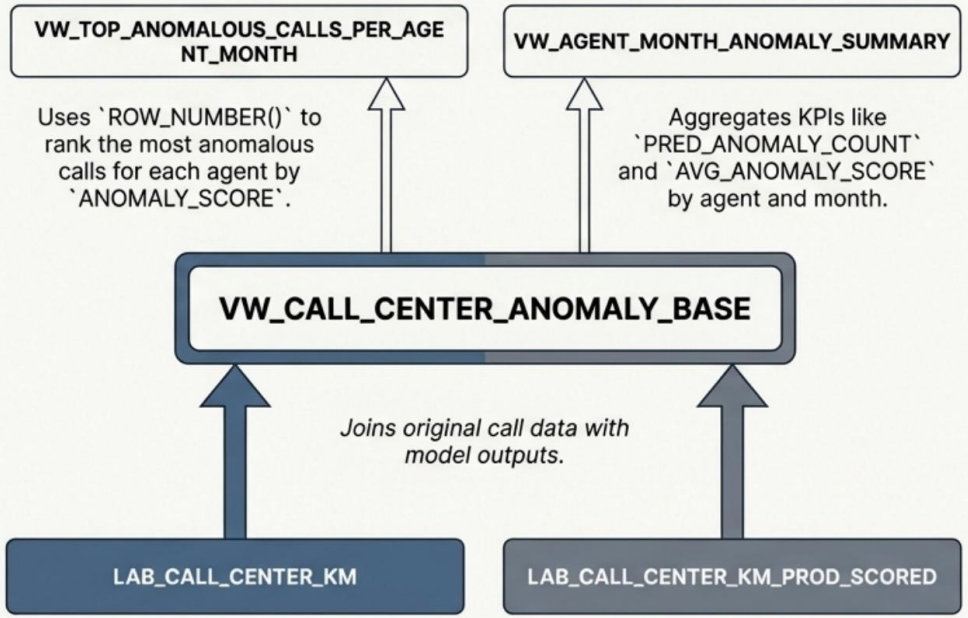
SQL ∨ as cell18

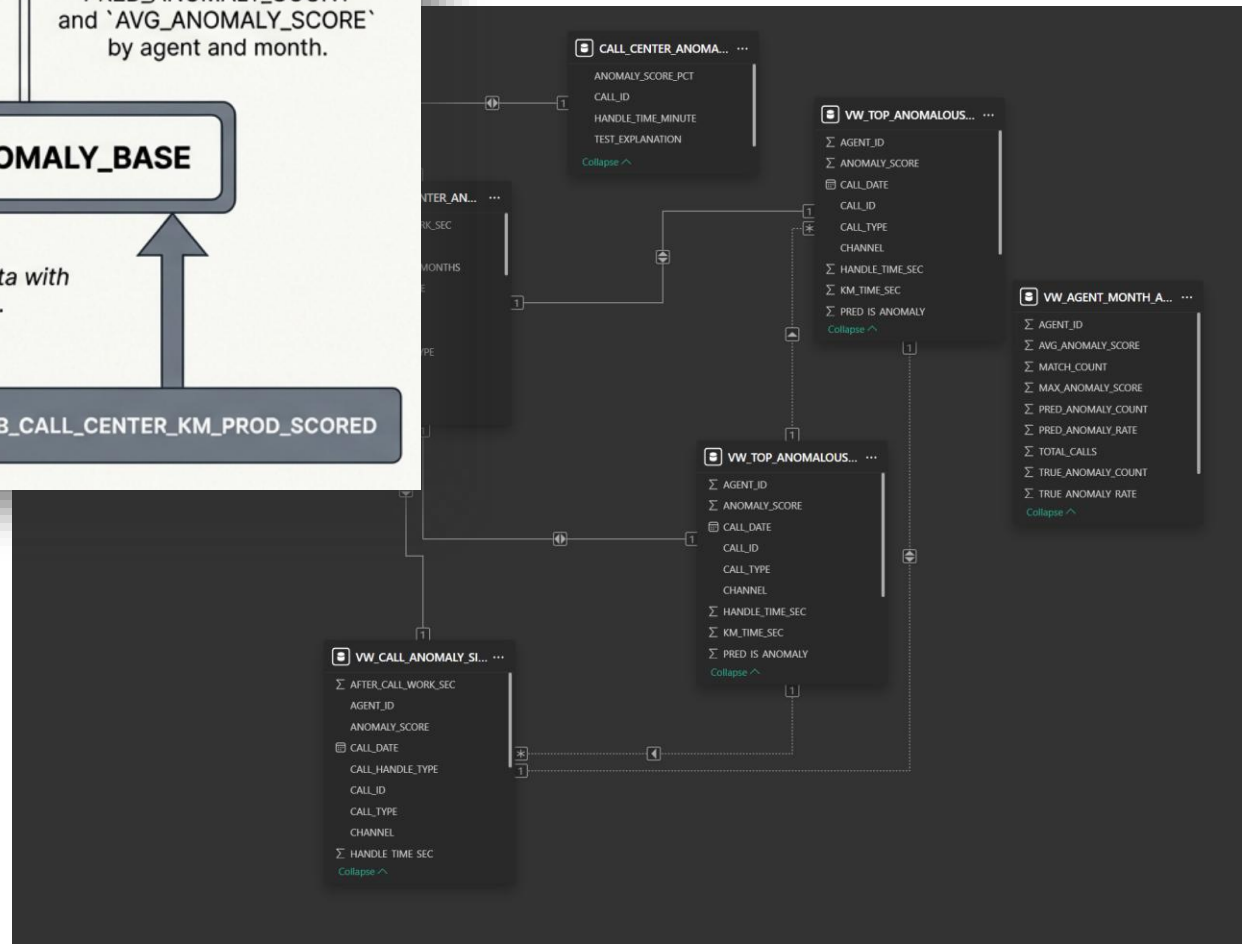
Building a Semantic Layer with SQL Views for Simplified Analytics

Objective

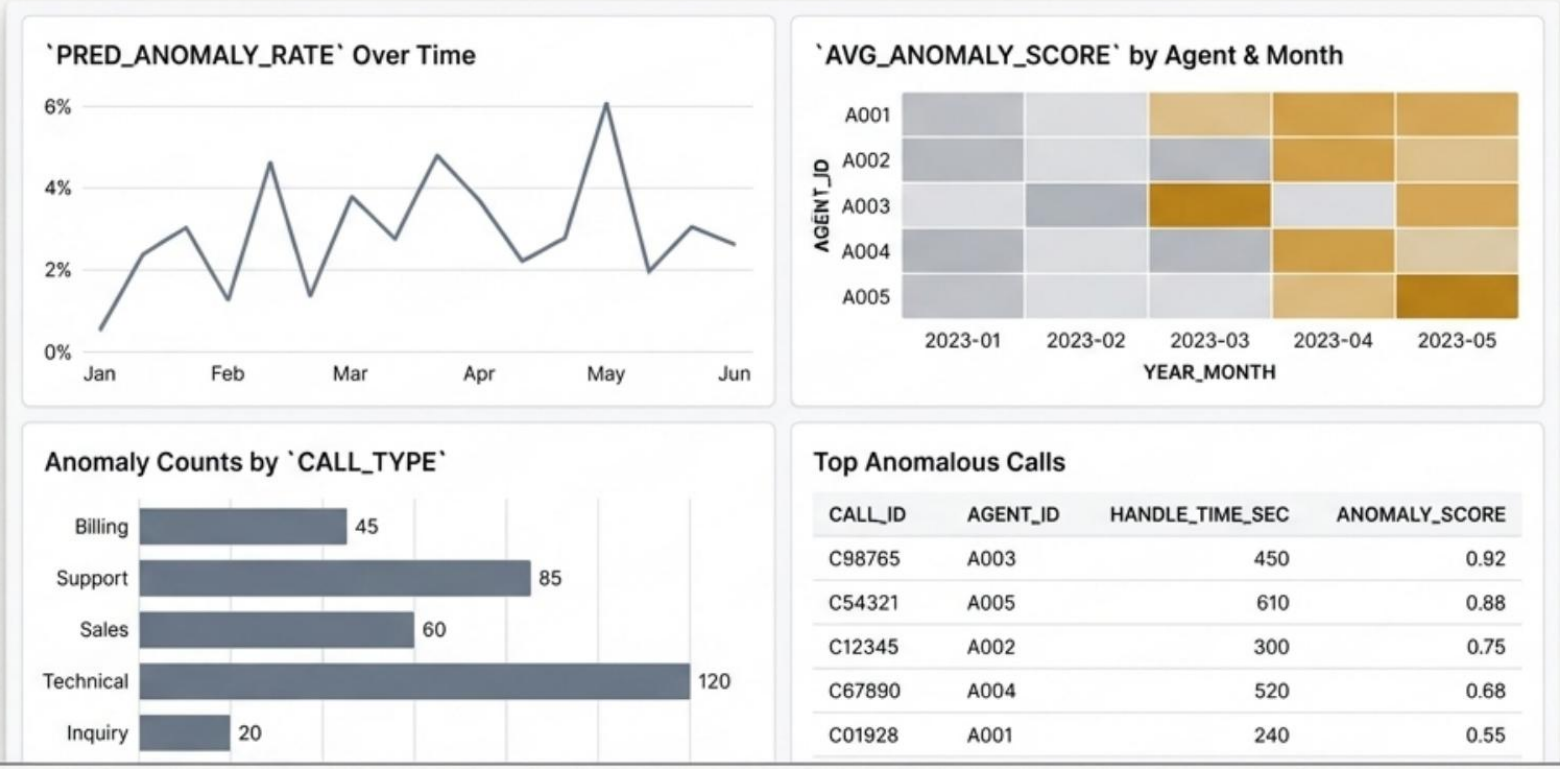
To provide a clean, unified data source for BI tools and downstream analysis without exposing the complexity of underlying tables.

Source Sans Pro Regular





Visualizing Anomalies: The Detection Dashboard



What we know

We can now precisely identify which agents, call types, and time periods are experiencing the most anomalies.

What we're missing

The quantitative data tells us *what* is anomalous, but not *why*. This requires manual investigation.

From ‘What’ to ‘Why’: Introducing Snowflake Cortex for AI Explanations

The Game-Changer

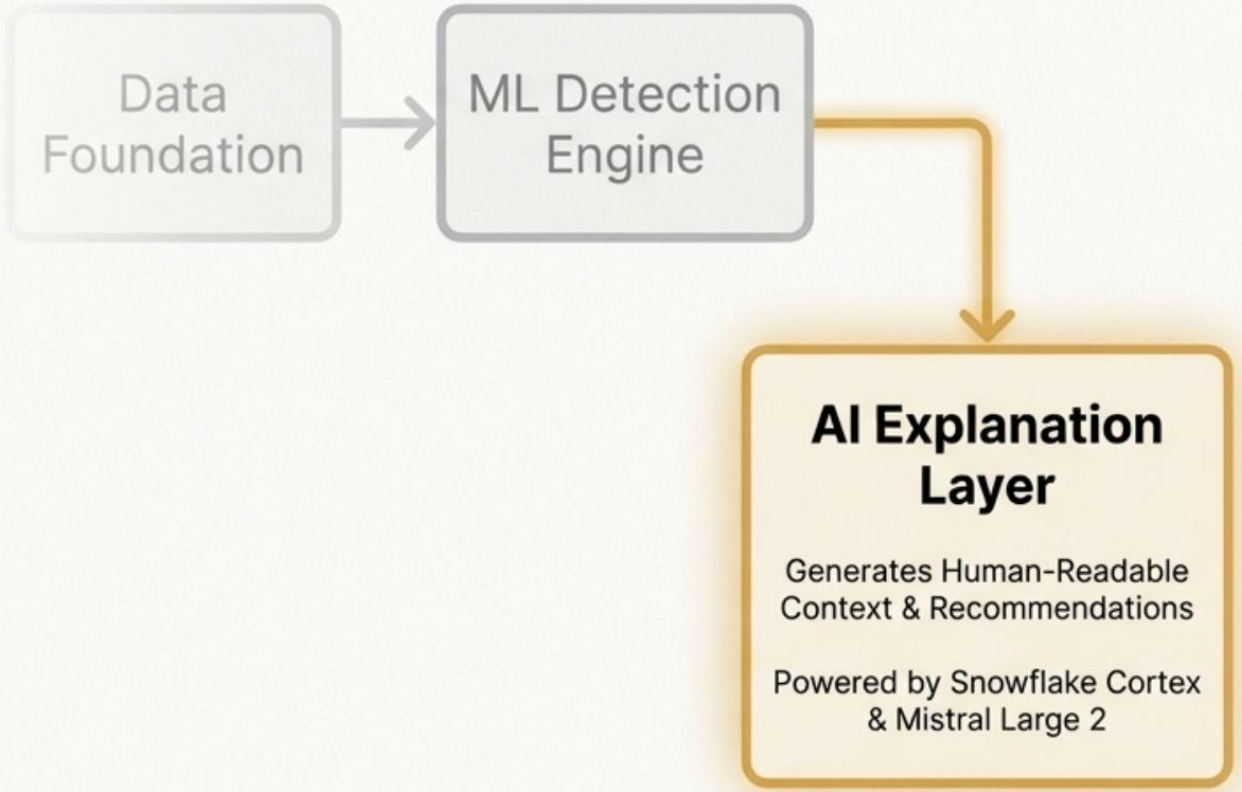
Snowflake Cortex provides built-in LLM capabilities directly via SQL functions, allowing us to generate narrative context without moving data out of the platform.

Core Function

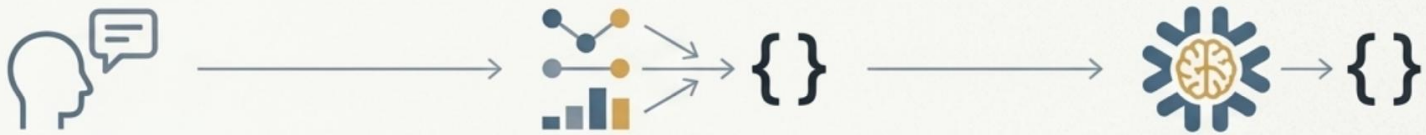
`SNOWFLAKE.CORTEX.COMPLETE` with the `` `mistral-large2` `` model.

Objective

Convert raw metrics and anomaly scores into plain-language, human-readable explanations and recommendations.



Generating Per-Call Explanations with a Cortex Complete Function



Step 1: Define a Persona

The prompt begins by instructing the model: "You are a senior call center performance analyst..."

Step 2: Construct a Payload

For each anomalous call in `VW_CALL_CENTER_ANOMALY_BASE`, we use `OBJECT_CONSTRUCT(...)` to create a JSON object containing key metrics (`HANDLE_TIME_SEC`, `KM_TIME_SEC`, `ANOMALY_SCORE`, etc.).

Step 3: Invoke the LLM

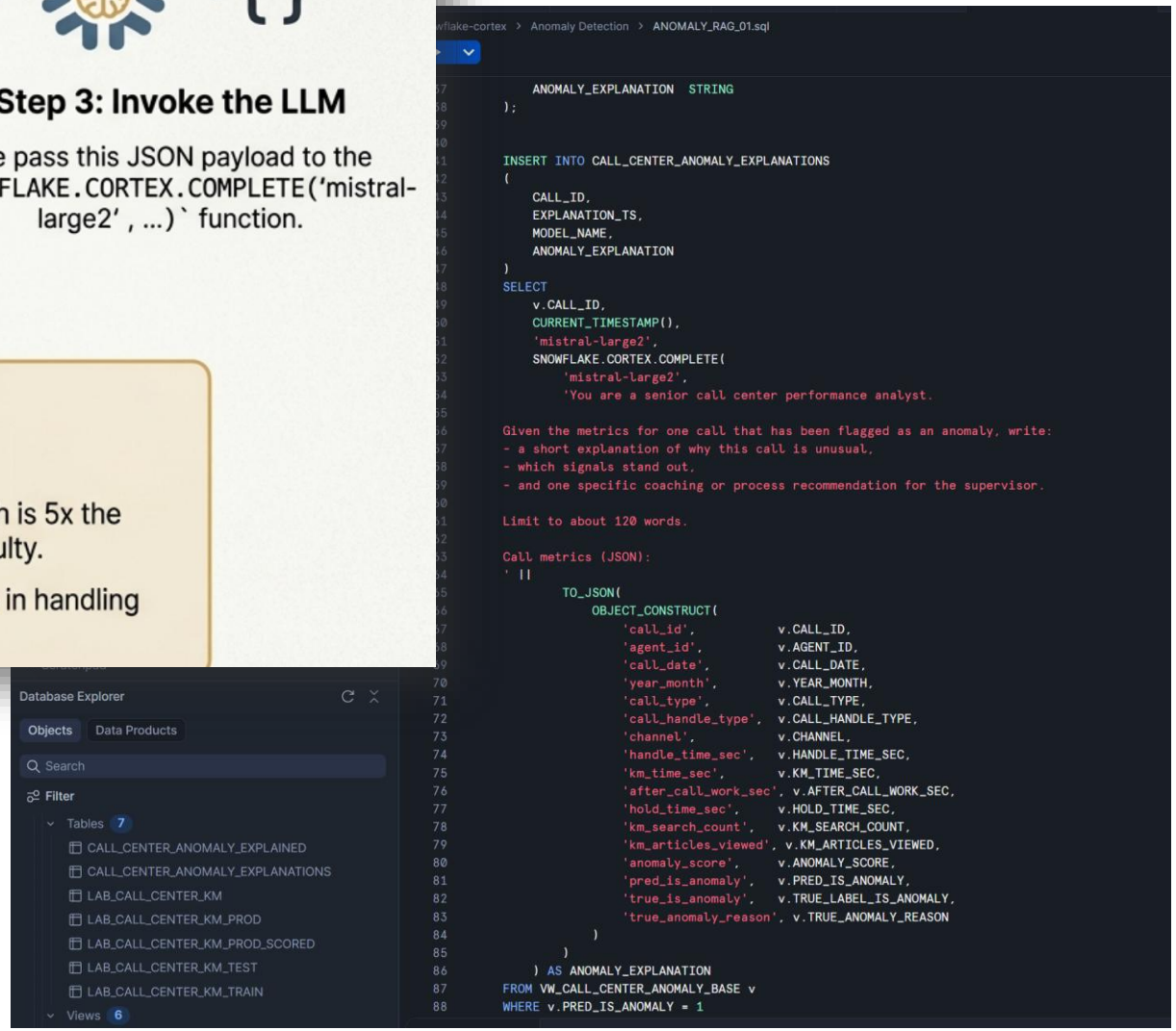
We pass this JSON payload to the `SNOWFLAKE.CORTEX.COMPLETE('mistral-large2', ...)` function.

Expected Output

Call ID: 12345-ABC

This call is anomalous due to an exceptionally high handle time (3200s), which is 5x the agent's average. This suggests a complex customer issue or procedural difficulty.

Coaching action: Review call recording with agent to identify knowledge gaps in handling this call type. Assess if KM articles need updating.

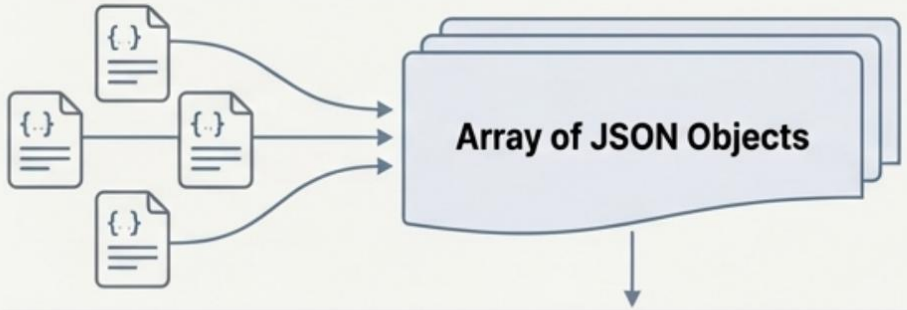


Uncovering Systemic Patterns with Aggregate-Level Summaries

Beyond Single Calls: We can use Cortex to analyze trends across multiple data points by feeding it aggregated data.

Example Use Case: Input

Individual agent summaries from 'VW_AGENT_MONTH_ANOMALY_SUMMARY'




```
[
  { "agent_id": 101, "month": "2023-11", "pred_anomaly_rate": 0.25, ... },
  { "agent_id": 102, "month": "2023-11", "pred_anomaly_rate": 0.05, ... },
  { "agent_id": 103, "month": "2023-11", "pred_anomaly_rate": 0.31, ... }
]
```

Cortex Task

Analyze the provided data to identify agents most at risk, describe common patterns across their anomalies, and recommend systemic interventions.

Cortex

Generated Narrative Summary

Key Observation:

Agents 101 and 103 consistently show the highest anomaly rates, primarily driven by long handle times on 'Billing Inquiry' calls.

Common Pattern:

These anomalies frequently occur during the last week of the month, suggesting a potential issue with end-of-month promotions or system performance.

Systemic Recommendation:

Propose a targeted training session for all agents on the 'Billing Inquiry' workflow and investigate the performance of the CRM tool during peak, end-of-month periods.

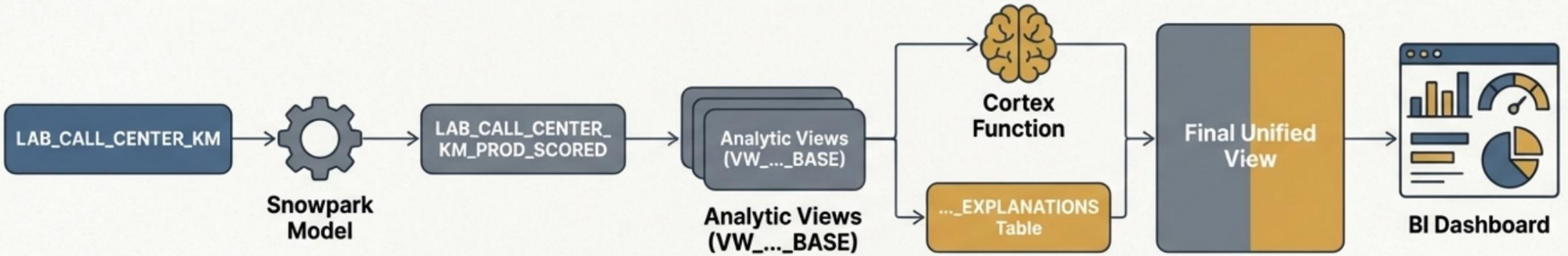
The Complete Solution: Integrating AI Narratives into the BI Layer

VW_CALL_CENTER_ANOMALY_BASE
(scores and metrics)

CALL_CENTER_ANOMALY_EXPLANATIONS
(Cortex-generated text)

VW_CALL_CENTER_ANOMALY_WITH_EXPLANATION
(Unified View)

The End-to-End Snowflake Workflow: From Data to Explained Insight



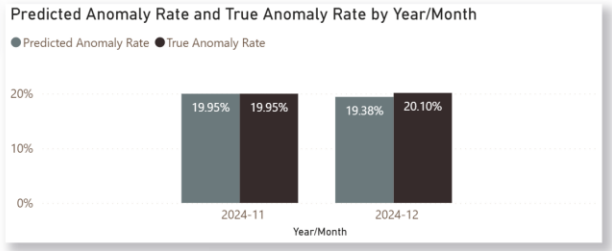
1. **Data Generation:** Create a labeled synthetic dataset in Snowflake.
2. **Model Training & Scoring:** Use Snowpark to build and apply an anomaly detection model.
3. **Semantic Layer:** Construct SQL views to simplify data for analytics.
4. **BI Visualization:** Identify what is anomalous in dashboards.
5. **AI Explanation:** Use Cortex to generate narratives explaining why.

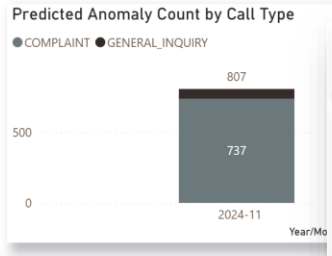
Prototype Dashboard

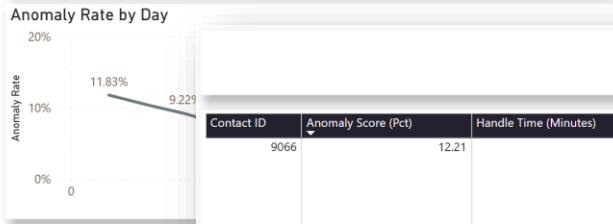
Anomaly Detection - Handle Time and Time in Knowledge Management Pages

Call Type
- Select all
- (Blank)
- COMPLAINT
- GENERAL_INQUIRY

16.39K	1603	9.78%	-0.05
Total Calls	Predicted Anomalies	Anomaly Rate	Average Anomaly Score







Anomaly Signals

Contact ID	Agent ID	Call Handle Type	Call Type	Channel	Handle Time	KM Pages Viewed	KM Search Count	KM View Time	Anomaly Flag	Year / Month
717	121	STANDARD	COMPLAINT	EMAIL	7,270	4	8	333	1	2024-11
964	385	STANDARD	COMPLAINT	PHONE	7,596	4	7	267	1	2024-11
1451	340	ABANDONED	COMPLAINT	PHONE	6,277	4	8	314	1	2024-12
2143	120	ABANDONED	COMPLAINT	EMAIL	1,838	4	7	289	1	2024-11
3215	3	ABANDONED	COMPLAINT	PHONE	9,410	4	5	283	1	2024-12
3403	310	ABANDONED	COMPLAINT	PHONE	4,931	4	8	345	1	2024-12
3404	213	ABANDONED	COMPLAINT	EMAIL	7,216	4	7	250	1	2024-11
4853	332	ABANDONED	COMPLAINT	EMAIL	7,663	3	7	329	1	2024-11
5076	332	STANDARD	COMPLAINT	PHONE	1,872	4	8	342	1	2024-12
5445	14	STANDARD	COMPLAINT	PHONE	7,385	4	7	282	1	2024-12
6075	469	ABANDONED	COMPLAINT	EMAIL	7,788	4	6	251	1	2024-12
					7,198	3	8	336	1	2024-11
					940	4	8	306	1	2024-12
					6,766	4	6	313	1	2024-12
					7,093	4	8	305	1	2024-11
					1,033	4	8	313	1	2024-12
					9,510	4	7	253	1	2024-11
					1,051	4	8	302	1	2024-12
					9,597	4	8	332	1	2024-11
					2,087	4	6	327	1	2024-11
					1,359	4	7	242	1	2024-12
					7,262	4	6	289	1	2024-11
					1,716	4	7	272	1	2024-11
					10,042	4	7	335	1	2024-12
					8,317	4	5	243	1	2024-12
					4,621	4	6	328	1	2024-12
					1,976	4	6	307	1	2024-12
					8,221	3	8	306	1	2024-11
					8,451	4	8	320	1	2024-12
					9,257	4	6	321	1	2024-11
					6,329	4	8	346	1	2024-12
					4,919	3	8	308	1	2024-12
					1,023	4	8	302	1	2024-12
					9,328	4	6	266	1	2024-11
					7,559	4	7	246	1	2024-11
					9,158	4	7	290	1	2024-12
					8,525	4	7	296	1	2024-12
					9,458	4	6	332	1	2024-11
					7,642	3	7	355	1	2024-11
					7,947	4	6	270	1	2024-12
					7,810	4	7	345	1	2024-11
					8,089	4	7	264	1	2024-11
					5,786	3	7	334	1	2024-12
					6,358	4	7	296	200	

Anomaly Explained by LLM

Contact ID	Anomaly Score (Pct)	Handle Time (Minutes)	Anomaly Expanation by "MISTAL-LARGE2" LLM
9066	12.21	159.94	Based on the provided metrics, here's a summary of the call: - **Call ID:** 9066 - **Anomaly Score:** 0.1221487905640415 (This score might indicate the call's deviation from the norm. The significance of this value would depend on the specific context and the model used to calculate it.) - **Handle Time:** 9596.56 seconds (This is approximately 2 hours and 40 minutes. This value represents the total time the call took, including hold time, talk time, and wrap-up time.) Without additional context (such as average handle time or what constitutes a high anomaly score), it's challenging to say whether these metrics indicate a problem or not. However, a handle time of nearly 2 hours and 40 minutes seems quite long for a typical call, which might warrant further investigation.
46342	11.04	150.20	Based on the provided metrics, here's a summary of the call: - **Call ID:** 46342 - **Anomaly Score:** 0.1104 (This score might indicate the call's deviation from the norm. The significance of this value would depend on the specific context and the model used to calculate it.) - **Handle Time:** Approximately 2 hours, 30 minutes, and 11.76 seconds (9011.76 seconds) Without additional context or benchmarks, it's challenging to determine if the handle time and anomaly score are within acceptable ranges. However, a handle time of over two hours might be considered quite long for many call center settings. The anomaly score may indicate that this call was somewhat different from typical calls, but without more information, it's difficult to say why or if this is a concern.
73271	11.00	140.06	Based on the provided metrics, here's a summary and some insights: 1. **Anomaly Score**: The anomaly score is approximately 0.11 (or 11%). This score indicates the likelihood of this call being anomalous compared to other calls. A higher score suggests a greater chance of the call being unusual. In this case, the score is relatively low, suggesting that the call is likely to be within normal parameters. 2. **Call ID**: The unique identifier for this call is 73271. 3. **Handle Time**: The handle time for this call is approximately 8403.84 seconds, which is about 2 hours and 20 minutes (140 minutes). This seems quite high for a single call and might warrant further investigation. Here are some potential follow-up questions or actions: - What's the average handle time for calls? If it's significantly lower than 140 minutes, then this call might be an outlier. - What was the context of this call? Was it a complex issue that required more time, or were there other factors at play? - How does the anomaly score for this call compare to others with similar handle times?

Value Realized and Future Roadmap

This solution demonstrates a complete, secure, and efficient workflow entirely within Snowflake, transforming raw data into not just detected anomalies, but understood and actionable insights.

Automation

Schedule daily scoring and explanation jobs using Snowflake tasks.

Model Expansion

Incorporate additional anomaly types, such as unusual KM usage patterns or abnormal call escalations.

Operational Integration

Connect the insights directly to workforce management or QA platforms to trigger and track follow-up actions.